WHAT IS CLAIMED IS:

1. A method for converting a user design for a programmable integrated circuit to a network of programmable logic blocks (PLBs), the method comprising:

combining non-strategic nodes in a logic cone with their predecessor nodes into PLBs only according to a first node merging group, wherein at least a portion of the user design is represented by the logic cone;

selecting one of a plurality of second node merging groups that can be used to combine a strategic node in the logic cone and at least one of its predecessor nodes into a PLB; and

combining the strategic node with at least one of its predecessor nodes into one PLB using the selected second node merging group.

2. The method according to 1 claim wherein selecting one of the second node merging groups further comprises:

determining whether the strategic node can be feasibly combined with at least one of its predecessor nodes into one PLB by considering limitations of the PLB's architecture.

3. The method defined in claim 1 wherein selecting one of the second node merging groups further comprises:

selecting the second merging group that generates the shortest depth of programmable logic blocks in a network resulting from a combination using that merging group.

4. The method defined in claim 1 wherein selecting one of the second node merging groups further comprises:

selecting one the second merging groups based on a cost metric that weighs area and depth of a programmable logic block network resulting from combinations using each second merging group.

5. The method defined in claim 1 wherein combining the non-strategic nodes with their predecessor nodes into the PLBs further comprises:

combining each non-strategic node and at least one of its predecessor nodes that have a maximum label value into one programmable logic block according to the first

merging group, if the combination is feasible based on an architecture of the programmable logic block; and

for each non-strategic node in the logic cone that cannot feasibly be combined with at least one of its predecessor nodes according to the first merging group, assigning a label value to the non-strategic node that equals the maximum value of its predecessor nodes plus one.

6. The method defined in claim 1 wherein combining the strategic node with at least one of its predecessor nodes into one programmable logic block using the selected second node merging group further comprises:

combining the strategic node with all of its non-boundary predecessor nodes into one programmable logic block.

7. The method defined in claim 1 wherein combining the strategic node with at least one of its predecessor nodes into one programmable logic block using the selected second node merging group further comprises:

combining the strategic node only with its immediately preceding fanin nodes.

8. The method defined in claim 1 wherein combining the strategic node with at least one of its predecessor nodes into one programmable logic block using the selected second node merging group further comprises:

combining the strategic node with its predecessor nodes in all directions until an input boundary node or a soft buffer boundary node is reached.

9. The method defined in claim 1 wherein combining the strategic node with at least one of its predecessor nodes into one programmable logic block using the selected second node merging group further comprises:

combining the strategic node with best merging groups of the strategic node's immediate fanin nodes.

10. The method defined in claim 1 wherein combining the strategic node with at least one of its predecessor nodes into one programmable logic block using the selected second node merging group further comprises:

combining the strategic node with its predecessor nodes that have the maximum label.

11. The method defined in claim 1 wherein the PLBs are macrocells and the programmable integrated circuit is a programmable logic device.

12. A computer system for converting a user design for a programmable integrated circuit into programmable logic blocks (PLBs), the computer system comprising:

code for combining non-strategic nodes in a logic cone and their predecessor nodes into PLBs according to only a first node merging group, wherein at least a portion of the user design is represented by the logic cone;

code for selecting one of a plurality of second node merging groups that can be used to combine a strategic node in the logic cone and at least one of its predecessor nodes into a PLB;

code for combining the strategic node with at least one of its predecessor nodes into one PLB according to the selected second node merging group; and

a computer readable media for storing the codes.

13. The computer system according to claim 12 wherein the code for selecting one of the plurality of second node merging groups further comprises:

code for identifying which of the second node merging groups can be feasibly placed inside a PLB based on constraints imposed by the architecture of the PLB.

14. The computer system according to claim 13 wherein the code for selecting one of the plurality of second node merging groups further comprises:

if more than one of the second node merging groups can be feasibly placed inside one PLB, the code for selecting selects one of the feasible second node merging groups based on a cost metric that weighs area and depth of a PLB network resulting from combinations using these second merging group.

15. The computer system according to claim 13 wherein the code for selecting one of the plurality of second node merging groups further comprises:

if more than one of the second node merging groups can be feasibly placed inside one PLB, the code for selecting selects the second node merging group that generates the least stages of PLBs in a network resulting from a combination using that second merging group.

16. The computer system according to claim 12 wherein the PLBs are macrocells and the programmable integrated circuit is a programmable logic device.

17. The computer system according to claim 12 wherein the second node merging groups include a merging group that combines a strategic node with all of its non-boundary predecessor nodes into one PLB.

18. The computer system according to claim 17 wherein the second node merging groups include a merging group that combines a strategic node only with its immediately preceding fanin nodes into one PLB.

19. The computer system according to claim 18 wherein the second node merging groups include a merging group that combines a strategic node with its predecessor nodes in all directions until an input boundary node or a soft buffer boundary node is reached.

20. The computer system according to claim 19 wherein the second node merging groups include a merging group that combines the strategic node with best merging groups of the strategic node's immediate fanin nodes.

21. The computer system according to claim 19 wherein the second node merging groups include a merging group that combines the strategic node with its predecessor nodes that have the maximum label.

22. The computer system according to claim 12 wherein the code for combining non-strategic nodes and their predecessor nodes into the PLBs combines each non-strategic node and at least one of its predecessor nodes that have a maximum label value into one PLB, if the combination is feasible based on an architecture of the PLB.

23. A method for converting a user design for a programmable integrated circuit into a network of programmable logic blocks (PLBs) for a programmable integrated circuit, the method comprising:

combining first nodes in a logic cone only with their predecessor nodes that have a maximum label into PLBs, wherein at least a portion of the user design is represented by the logic cone;

selecting one of a plurality of node merging groups that can be used to combine a second node in the logic cone and at least one of its predecessor nodes into a PLB; and

combining the second node with at least one of its predecessor nodes into one PLB using the selected node merging group,

wherein the second node is connected directly to an output boundary node of the logic cone and does not fanout to other non-boundary nodes, and the first nodes are the remaining non-boundary nodes in the logic cone.